Introduction To Unix Editing Tools

Steve Gallo and Andrew Bruno

Center for Computational Research
University at Buffalo
State University of New York

CCR Summer Seminar Series, 2012

Introduction

There are many editors to choose from, both graphical and text-based.

- VIM
- Emacs
- Gedit
- Nano
- Sublime Text 2

... and the list goes on

http://en.wikipedia.org/wiki/List_of_text_editors

Review – Environment Variables

Environment variables control various aspects of your environment and applications. These can be set in your login scripts, by applications, or on the command line

bash: export EDITOR=/usr/bin/emacs
csh: setenv VISUAL /usr/bin/vi

Useful environment variables
- **EDITOR** and **VISUAL** – set the editor used by various applications
- **PATH** – A colon-separated list of directories searched for applications to run (e.g., PATH=/bin:/usr/bin:/usr/local/bin)
- **HOME** – Your home directory on the machine

Review – Job Control

Unix supports the notion of foreground and background processes. When you run an application (i.e., a job) from the command line it typically runs in the foreground, but you can also run it in the background using the "&" operator.

```
# Run emacs in the foreground. The shell waits for you to exit emacs.
$ emacs test.c

# Run emacs in the background - you are returned to the shell.
$ emacs test.c &
$
```

Use C-z to suspend an job and return to the shell. Type "bg" to continue running the job in the background or "fg" to return to the job. Useful for suspending a job to perform a task and then returning to it. For example,

```
# Use vi to edit a file but you forgot to do something and don't want to lose your
place!
$ vi test.c
   ^Z
# The less command is now suspended
$ cp /some/file .
$ jobs
[1]+  Stopped                 vi test.c
$ fg
# Now you're back in vi where you left off
```

Editors

Emacs and Vim are commonly installed on Unix systems.

Topics
- Opening and saving files, navigating, selection, cut-and-paste
- Auto-completion, syntax highlighting, modes
- Advanced editing: buffers, regions
- Exercises

Advanced Topics for Programmers
- Customizations
- Remote editing

Emacs

Guided Tour

http://www.gnu.org/software/emacs/tour/

Online Manual

http://www.gnu.org/software/emacs/manual/html_mono/emacs.html

Need a better editor for Windows? Use Emacs!

http://ftp.gnu.org/gnu/emacs/windows/

Help from within Emacs
M-x info
M-x help

Emacs – Notation

M-x is used to run a command

    Stands for Meta-x (modifier) and means press the escape key followed by "x"

    For example, **M-x tetris** starts a tetris game

    Use tab twice to see all commands that start with a string or all available options!

```
Click <mouse-2> on a completion to select it.
In this buffer, type RET to select the completion near point.

Possible completions are:
cua--prefix-copy-handler          cua--prefix-cut-handler
cua--prefix-override-handler      cua--prefix-repeat-handler
cua--shift-control-c-prefix       cua--shift-control-x-prefix
cua-cancel                        cua-copy-region
cua-cut-region                    cua-debug
cua-delete-region                 cua-exchange-point-and-mark
cua-help-for-region               cua-mode
cua-paste                         cua-paste-pop
cua-repeat-replace-region         cua-replace-region
cua-scroll-down                   cua-scroll-up
cua-selection-mode                cua-set-mark
cua-set-rectangle-mark            cua-toggle-global-mark

-U:%%-  *Completions*   All (1,0)      (Completion List)----Wed Jun 20 10:29AM 0.93---
   M-x cua-
```

C-x is a shortcut for a command, typically followed by a command key

    Press the control key at the same time as "x" typically followed by a shortcut key

    For example, **C-x C-s** will save your file

Emacs – Entering and Exiting Files

Two modes of operation: In its own window (graphical mode) or within a terminal (text mode - useful for editing files remotely).

```
# Run emacs in "no window" mode within a terminal
$ emacs -nw GoogleScholar.php

# Run emacs in its own window (in the background)
$ emacs GoogleScholar.php &
$
```

How do I exit Emacs and save my work?

Key	Description
C-x C-s	Save a buffer to its file
C-x s	Save all modified buffers
C-x C-w	Save as
C-x C-c	Exit (asks to save any modified buffers)

The Emacs Window
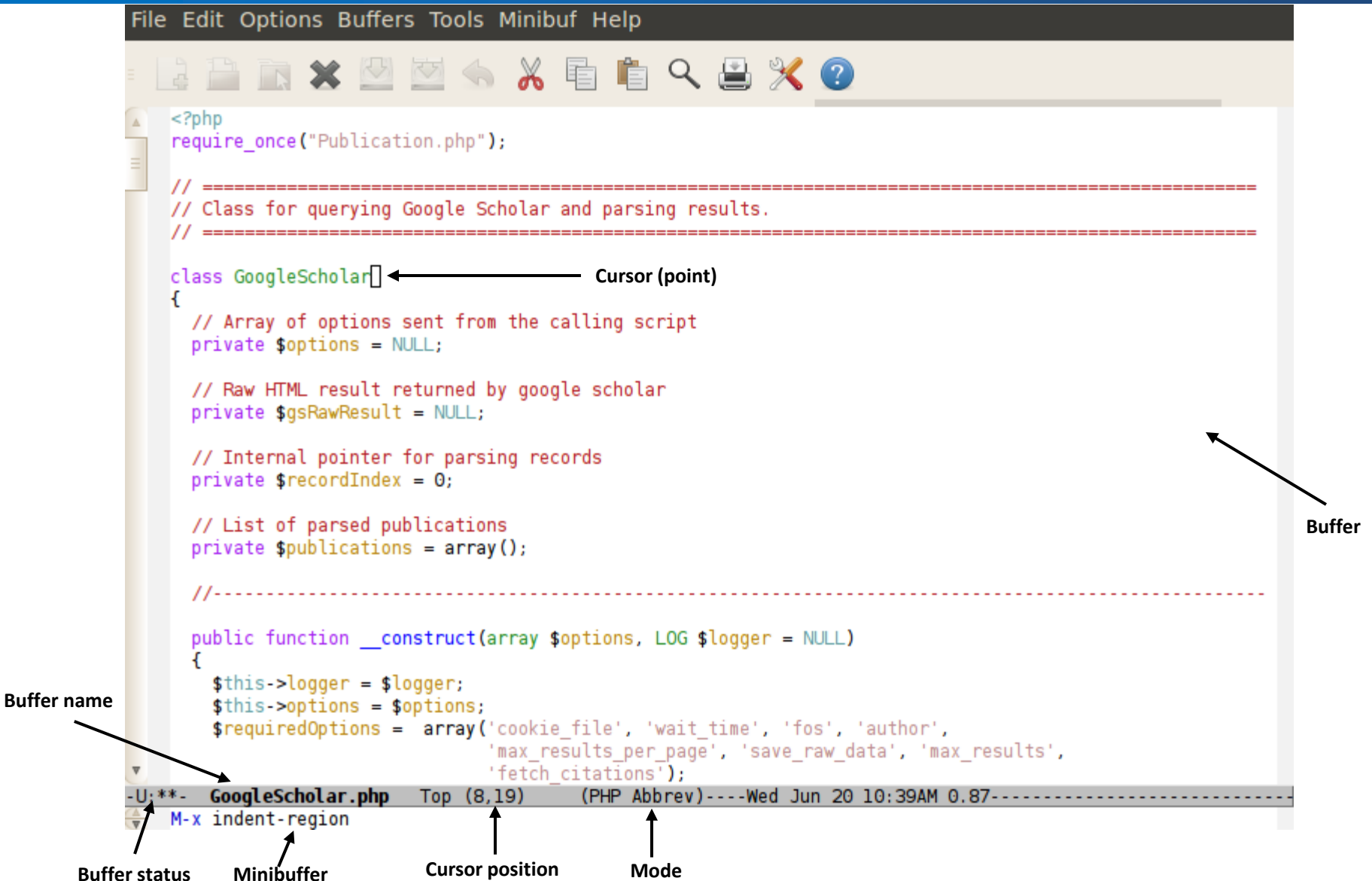
Emacs – Buffers and Regions

Buffers

- Any text being edited (such as a file) is contained in a buffer
- A buffer doesn't have to contain data from a file!
- Every buffer has a name - typically the name of the file
- You can switch between buffers (using C-leftmouse in graphical mode or C-x b)
- You can split buffers to view several on the screen at once

Regions

- A region is a marked area within a buffer (e.g., a selection)
- Many operations can be performed on an entire buffer or a region
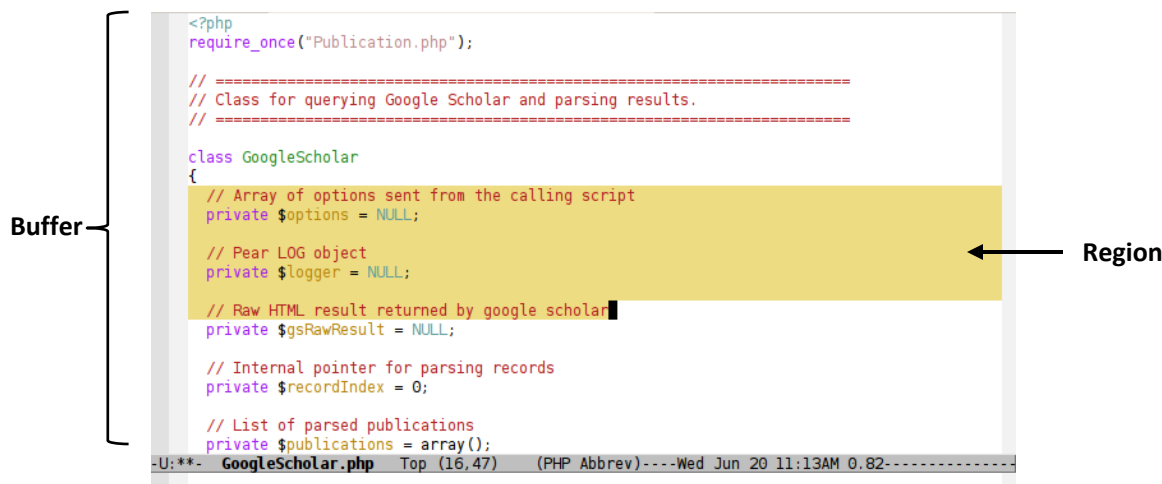
Emacs – Moving Around

The arrow keys let you move around easily within Emacs, but to be more productive you can also use command keys!

Key	Description
arrows	up, down, left, right
C-p , **C-n** , **C-b** , **C-f**	up, down, left, right
C-arrow	Same as arrow but moves a whole word
C-a (or Home key)	Move to the beginning of the line (also works in the shell)
C-e (or End key)	Move to the end of the line (also works in the shell)
C-v and **M-v**	Page down and Page up
M-g g	Go to line (I aliased this to **M-:** and yes I stole it from vi!)
M-< and **M->**	Move to the top of the buffer and move to the bottom

Why use **C-a** instead of the home key or **C-f** instead of the arrow? Isn't that crazy?

No! If you have one hand on the keyboard and one on the mouse every time you take your hand off of the mouse to click a key it slows you down. The same is true for copy and paste – if you take your hands off the keyboard to use the mouse menu it slows you down. Learning shortcuts can make you more productive.

Emacs – Cut and Paste

Cut and paste in emacs is called "kill" and "yank". Killing text puts that text into a buffer that you can then yank back out.

Key	Description
C-k	Kill a line from the cursor to the end of the line
C-y	Yank the contents of the kill buffer back into the document
C-d	Delete a single character under the cursor
Backspace	Delete the character before the cursor
M-x cua-mode	Don't like **C-k** and **C-y**? Set Common User Access (CUA) mode! **C-x**, **C-c**, **C-v**, and **C-z** become cut, copy, paste, and undo.

Cut and paste also works with regions. Select a region using the mouse in graphical mode or command keys. Move the cursor to another location and kill the text in the region. Regions are useful – more to come.

Key	Description
C-<space>	Set the mark at the cursor
C-w	Kill from the mark to the cursor

Emacs – Working With Rectangles

We often need to operate on rectangular regions of text within a document. Delete a column, move it elsewhere, clear a column, etc. To select a rectangle, set the mark at one corner and move the cursor to the opposite corner.

Key	Description
C-x r k	Kill the rectangle
C-x r y	Yank a killed rectangle
C-x r o	Insert blank space into the marked rectangle, moving the current contents to the right.
C-x r c	Clear the rectangle leaving a blank space

Emacs – Opening Files

Emacs can open several files at once, either from the command-line or within Emacs itself.

```
# Open two files.  The files will be displayed in a split window
$ emacs GoogleScholar.php sample.txt &
```

When opening a file from within emacs you can use tab-completion to browse for a file or enter `dired` (directory edit) mode to select a file.

Key	Description
C-x C-f	Browse or open a file
C-x C-r	Browse or open a file read-only
C-x k	Kill the current buffer and close the file

Emacs – Viewing Multiple Files

You'll often be working on multiple files and will want to view them at the same time. You can open multiple buffers or windows (frames), split the window horizontally, or split the window vertically.

Key	Description
C-x 2	Split the current buffer **horizontally** into two
C-x 3	Split the current buffer **vertically** into two
C-x o	Switch to **o**ther buffer (cycles through multiple buffers)
C-x 0	Delete the current window
C-x 1	Delete all windows EXCEPT the current window
C-x 5 2	Open a new window (frame in Emacs-speak)

Emacs – Modes

Every buffer in Emacs has a mode associated with it based on the type of data being displayed. Modes modify Emacs' behavior to support common operations on the type of file being edited and sets the mode automatically based on the file extension.

Indentation, highlighting of expressions and key words, and other language-specific features are affected by the mode.

Modes are highly customizable (but that is an advanced topic…)

Key	Description
M-x php-mode	Automatically indents code blocks, adds highlighting
M-x text-mode	Sets text mode. Automatically wraps text at 80 characters (can be changed)
M-x auto-fill	You can turn off auto fill mode while in text mode. This can be useful for editing data files.

Emacs – Useful Features

Symbol completion while programming

```
scholarL M-<tab>
scholarList
scholarListOfPublications
…
```

Entering multiple characters at once

```
<esc> 80 -
--------------------------------------------------------------------------
```

Adaptive fill mode

```
- Create Simon & Carol an iRods account
- Multiple zones connected to 2 different irods instances? (split Michigan
  public/private)
- Search fields
- Rule to read metadata from imported file and add to metadata
```

```
- Create Simon & Carol an iRods account Multiple zones connected to 2 different
- irods instances? (split Michigan public/private) Search fields Rule to read
- metadata from imported file and add to metadata
```

Undo

Key	Description
M-<tab>	Cycle through possible symbol completions
<esc> <number> <char>	Insert <number> of <char>
M-x filladapt-mode	Enable adaptive fill mode
C-x u	Undo recent editing

VIM

"Emacs is a hideous monstrosity, but a functional one. On the other hand, vi is a masterpiece of elegance. Sort of like a Swiss Army knife versus a rapier."

--The Cult of vi

History of vi

- vi was originally written by Bill Joy in 1976 while at Berkeley
- Bill Joy is a co-founder of Sun Microsystems and creator of csh among many others
- vi is based off ex and ed, the original line editor for Unix written by Ken Thompson

History of vi..

- vi has many ports and clones
- VIM is a vi clone written by Bram Moolenaar that adds many advanced features
- VIM - Vi IMproved
- First public release was in 1991
- VIM is included with almost every Linux distribution
- Also has a GUI version called gVim

VIM Basics

VIM is a modal editor

- Three basic modes of operation: **insert**, **visual**, and **command**
- By default, VIM is run in command mode. Each mode is accessed by pressing a key and **ESC** returns to command mode.
- To insert text you need to switch to insert mode by pressing **i**
- Visual mode allows you to highlight text. To enter visual mode press **v**
- To view the VIM user manual type **:help** or :help [command]
- Even will explain the meaning of life **:help 42**

VIM – Entering and Exiting Files

Type **vim** (or just **vi**) in the shell to start. When vim starts you'll be in normal (command) mode.

```
# Run vim
$ vim hello-world.c
```

How do I exit **vim** and save my work?

Command	Description
:w	Save a buffer to its file
:wa	Save all modified buffers
:w newfile	Save as
:q	Exit
:q!	Exit without saving changes

VIM – Moving Around

While in command mode:

Key	Description
k,j,l,h	up, down, left, right
arrows	up, down, left, right
w	Same as above but moves a whole word
0 (zero)	Move to the beginning of the line
$	Move to the end of the line
C-f and **C-b**	Page down and Page up
12g or :12<Enter>	Go to line, (example go to line 12)
gg G	Move to the top of the buffer and move to the bottom
ESC	Return to command mode
/ ?	Search forward and Search reverse
yy p	Copy entire line and paste

Try using the "home row" for moving around. It allows you to navigate your file without moving your hand over to the arrow keys.

```
          k
    h         l
          j
```
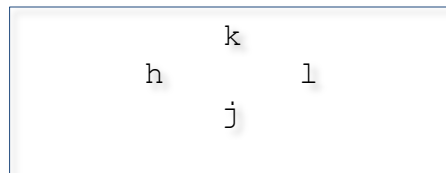
VIM Resources

- VIM – http://www.vim.org
- Cheat Sheets – http://www.viemu.com/a_vi_vim_graphical_cheat_sheet_tutorial.html